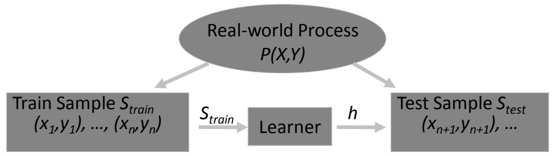
Empirical Risk Minimization, Model Selection, and Model Assessment

CS6780 – Advanced Machine Learning
Spring 2015

Thorsten Joachims
Cornell University

Reading:
Murphy 5.7-5.7.2.4, 6.5-6.5.3.1
Dietterich, T. G., (1998). Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. Neural Computation, 10 (7) 1895-1924.
(http://sci2s.ugr.es/keel/pdf/algorithm/articulo/dietterich1998.pdf

Learning as Prediction: Batch Learning Model



- Definition: A particular Instance of a Learning Problem is described by a probability distribution $P(X,Y)$.
- Definition: Any Example (X_i, Y_i) is a random variable that is independently identically distributed according to $P(X,Y)$.

Training / Validation / Test Sample

- Definition: A Training / Test / Validation Sample $S = ((x_1, y_1), \ldots, (x_n, y_n))$ is drawn iid from $P(X,Y)$.

$$P\left(S = ((x_1,y_1), \ldots, (x_n,y_n))\right) = \prod_{i=1}^{n} P(X_i = x_i, Y_i = y_i)$$

Risk

- Definition: The Risk / Prediction Error / True Error / Generalization Error of a hypothesis h for a learning task $P(X,Y)$ is

$$Err_P(h) = \sum_{x,y} \Delta\big(y, h(x)\big)\, P(X = x, Y = y)$$

- Definition: The Loss Function $\Delta(y, \hat{y}) \in \Re$ measures the quality of prediction $\hat{y}$ if the true label is y.

Empirical Risk

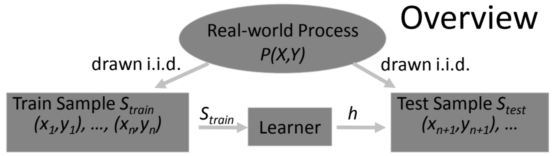
- Definition: The Empirical Risk / Error of hypothesis h on sample

$$S = ((x_1, y_1), \ldots, (x_n, y_n))$$

is

$$Err_S(h) = \sum_{i=1}^{n} \Delta(y_i, h(x_i))$$

Learning as Prediction Overview



- Goal: Find h with small prediction error $Err_P(h)$ with respect to $P(X,Y)$.

- Training Error: Error $Err_{S_{train}}(h)$ on training sample.
- Test Error: Error $Err_{S_{test}}(h)$ on test sample is an estimate of $Err_P(h)$.

Bayes Risk

- Given knowledge of P(X,Y), the true error of the best possible h is

$$Err_P(h_{bayes}) = E_{x\sim P(X)}[\min_{y\in Y}(1 - P(Y = y|X = x))]$$

for the 0/1 loss.

Three Roadmaps for Designing ML Methods

- Generative Model:
  →Learn $P(X, Y)$ from training sample.
- Discriminative Conditional Model:
  →Learn $P(Y|X)$ from training sample.
- Discriminative ERM Model:
  → Learn h directly from training sample.

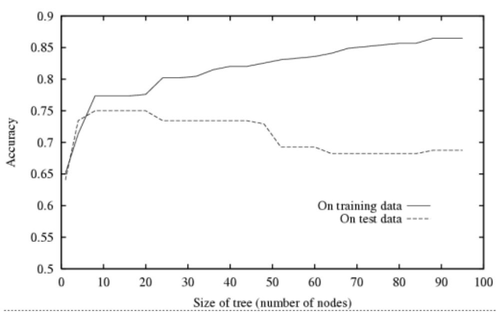
Empirical Risk Minimization

- Definition [ERM Principle]: Given a training sample $S = ((x_1, y_1), \dots, (x_n, y_n)$ and a hypothesis space H, select the rule $h^{ERM} \in H$ that minimizes the empirical risk (i.e. training error) on S

$$h^{ERM} = \min_{h\in H} \sum_{i=1}^{n} \Delta(y_i, h(y_i))$$

MODEL SELECTION

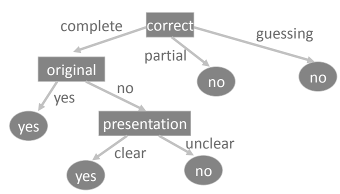
Overfitting



- Note: Accuracy = 1.0-Error

[Mitchell]

Decision Tree Example: revisited

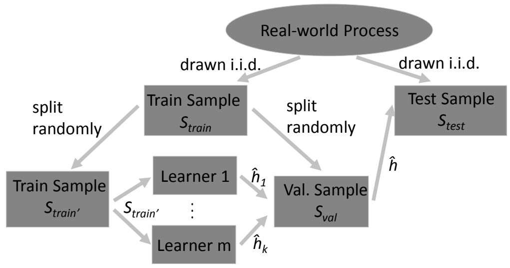


C O P	A^+
$\vec{x}_1 = (\text{c, y, c})$	$y_1 = +1$
$\vec{x}_2 = (\text{c, n, u})$	$y_2 = -1$
$\vec{x}_3 = (\text{c, y, u})$	$y_3 = +1$
$\vec{x}_4 = (\text{c, n, c})$	$y_4 = +1$
$\vec{x}_5 = (\text{p, y, c})$	$y_5 = -1$
$\vec{x}_6 = (\text{g, y, c})$	$y_6 = -1$
$\vec{x}_7 = (\text{c, y, c})$	$y_7 = +1$
$\vec{x}_8 = (\text{c, y, u})$	$y_8 = +1$
$\vec{x}_9 = (\text{p, y, c})$	$y_9 = -1$
$\vec{x}_{10} = (\text{c, y, c})$	$y_{10} = +1$

Controlling Overfitting in Decision Trees

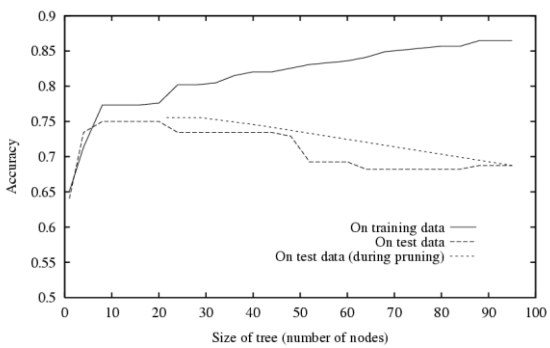
- Early Stopping: Stop growing the tree and introduce leaf when splitting no longer "reliable".
  - Restrict size of tree (e.g., number of nodes, depth)
  - Minimum number of examples in node
  - Threshold on splitting criterion
- Post Pruning: Grow full tree, then simplify.
  - Reduced-error tree pruning
  - Rule post-pruning

Model Selection via Validation Sample



- **Training:** Run learning algorithm m times (e.g. different parameters).
- **Validation Error:** Errors $Err_{S_{val}}(\hat{h}_i)$ is an estimates of $Err_P(\hat{h}_i)$ for each h_i.
- **Selection**: Use h_i with min $Err_{S_{val}}(\hat{h}_i)$ for prediction on test examples.

Reduced-Error Pruning

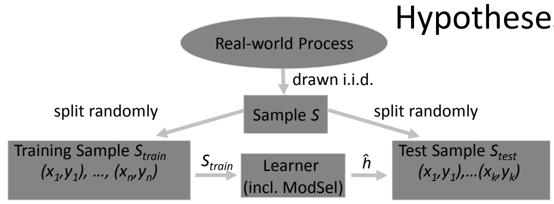


Text Classification Example "Corporate Acquisitions" Results

- Unpruned Tree (ID3 Algorithm):
  - Size: 437 nodes Training Error: 0.0% Test Error: 11.0%
- Early Stopping Tree (ID3 Algorithm):
  - Size: 299 nodes Training Error: 2.6% Test Error: 9.8%
- Reduced-Error Pruning (C4.5 Algorithm):
  - Size: 167 nodes Training Error: 4.0% Test Error: 10.8%
- Rule Post-Pruning (C4.5 Algorithm):
  - Size: 164 tests Training Error: 3.1% Test Error: 10.3%
  - Examples of rules
    - IF vs = 1 THEN - [99.4%]
    - IF vs = 0 & export = 0 & takeover = 1 THEN + [93.6%]

MODEL ASSESSMENT

Evaluating Learned Hypotheses

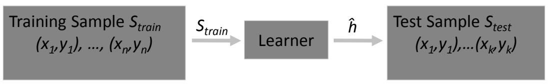


- Goal: Find h with small prediction error $Err_P(h)$ over $P(X,Y)$.
- Question: How good is $Err_P(\hat{h})$ of $\hat{h}$ found on training sample S_{train}.

- **Training Error:** Error $Err_{S_{train}}(\hat{h})$ on training sample.
- **Test Error:** Error $Err_{S_{test}}(\hat{h})$ is an estimate of $Err_P(\hat{h})$.

What is the True Error of a Hypothesis?

- Given
  - Sample of labeled instances S
  - Learning Algorithm A
- Setup
  - Partition S randomly into S_{train} and S_{test}
  - Train learning algorithm A on Strain, result is $\hat{h}$.
  - Apply $\hat{h}$ to S_{test} and compare predictions against true labels.
- Test
  - Error on test sample $Err_{S_{test}}(\hat{h})$ is estimate of true error $Err_P(\hat{h})$.
  - Compute confidence interval.



Binomial Distribution

- The probability of observing x heads (i.e. errors) in a sample of n independent coin tosses (i.e. examples), where in each toss the probability of heads (i.e. making an error) is p, is

$$P(X=x|p,n) = \frac{n!}{x!(n-x)!}p^x(1-p)^{n-x}$$

- Normal approximation: For $np(1-p)>=5$ the binomial can be approximated by the normal distribution with
  - Expected value: $E(X)=np$ Variance: $Var(X)=np(1-p)$
  - With probability δ, the observation x falls in the interval

$$E(X) \pm z_\delta \sqrt{Var(X)}$$

δ	50%	68%	80%	90%	95%	98%	99%
z_δ	0.67	1.00	1.28	1.64	1.96	2.33	2.58

Is Rule h_1 More Accurate than h_2?

- Given
  - Sample of labeled instances S
  - Learning Algorithms A_1 and A_2
- Setup
  - Partition S randomly into S_{train} and S_{test}
  - Train learning algorithms A_1 *and* A_2 on S_{train}, result are $\hat{h}_1$ and $\hat{h}_2$.
  - Apply $\hat{h}_1$ and $\hat{h}_2$ to S_{test} and compute $Err_{S_{test}}(\hat{h}_1)$ and $Err_{S_{test}}(\hat{h}_2)$.
- Test
  - Decide, if $Err_P(\hat{h}_1) \neq Err_P(\hat{h}_2)$?
  - Null Hypothesis: $Err_{S_{test}}(\hat{h}_1)$ and $Err_{S_{test}}(\hat{h}_2)$ come from binomial distributions with same p.

    → Binomial Sign Test (McNemar's Test)

Is Learning Algorithm A_1 better than A_2?

- Given
  - k samples $S_1 \ldots S_k$ of labeled instances, all i.i.d. from P(X,Y).
  - Learning Algorithms A_1 and A_2
- Setup
  - For i from 1 to k
    - Partition S_i randomly into S_{train} and S_{test}
    - Train learning algorithms A_1 *and* A_2 on S_{train}, result are $\hat{h}_1$ and $\hat{h}_2$.
    - Apply $\hat{h}_1$ and $\hat{h}_2$ to S_{test} and compute $Err_{S_{test}}(\hat{h}_1)$ and $Err_{S_{test}}(\hat{h}_2)$.
- Test
  - Decide, if $E_S(Err_P(A_1(S_{train}))) \neq E_S(Err_P(A_2(S_{train})))$?
  - Null Hypothesis: $Err_{S_{test}}(A_1(S_{train}))$ and $Err_{S_{test}}(A_2(S_{train}))$ come from same distribution over samples S.

    → t-Test or Wilcoxon Signed-Rank Test

Approximation via K-fold Cross Validation

- Given
  - Sample of labeled instances S
  - Learning Algorithms A_1 and A_2
- Compute
  - Randomly partition S into k equally sized subsets $S_1 \ldots S_k$
  - For i from 1 to k
    - Train A_1 and A_2 on $S_1 \ldots S_{i-1} S_{i+1} \ldots S_k$ and get $\hat{h}_1$ and $\hat{h}_2$.
    - Apply $\hat{h}_1$ and $\hat{h}_2$ to S_i and compute $Err_{S_i}(\hat{h}_1)$ and $Err_{S_i}(\hat{h}_2)$.
- Estimate
  - Average $Err_{S_i}(\hat{h}_1)$ is estimate of $E_S(Err_P(A_1(S_{train})))$
  - Average $Err_{S_i}(\hat{h}_2)$ is estimate of $E_S(Err_P(A_2(S_{train})))$
  - Count how often $Err_{S_i}(\hat{h}_1) > Err_{S_i}(\hat{h}_2)$ and $Err_{S_i}(\hat{h}_1) < Err_{S_i}(\hat{h}_2)$